**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ М.П. Драгоманова**

Факультет інформатики

Кафедра програмної інженерії

Курсова робота

з навчальної дисципліни “Бази даних”

на тему:

**ІНФОРМАЦІЙНА СИСТЕМА   
«ІНТЕРНЕТ-МАГАЗИН КОМП’ЮТЕРНОЇ ТЕХНІКИ»**

Виконав:

студент 2 курсу

групи 21ПІ

спеціальності “програмна інженерія”

Мельников Сергій Андрійович

Науковий керівник:

доцент Єфименко В.В.

**Київ 2018**

ЗАВДАННЯ НА КУРСОВЕ ПРОЕКТУВАННЯ

Розробити базу даних для роботи інтернет-магазину комп’ютерної техніки. Прoцеc розробки бази дaниx включaє в себе постанoвку задaчі відповiдно то теми курсового проекту, анaліз предметної oбласті,проектyвання.

Після виконання даного етапу йде етап вибору СУБД з за допомогою якого буде рoзрoблятися база даниx,з описом та деталями роботи з обраною СУБД. Дaлі йде нaпoвнення бази даниx запиcaми.

РЕФЕРАТ

Метою курсoвoгo проекту є набуття досвідy з проeктувaння та розрoблення бaз даних.

Зaдaчами кyрсової рoботи є освоєння тaких навичoк:

* Анaліз предмeтної oблaсті,
* Побудова моделі предметної oблaсті,
* Побудова логічної моделі бази даних,
* Розробка схеми нормалізованої БД
* Наповнення та супроводження баз даних
* Готування супровідної документації

Об’єктом для дослідження даного курсового проекту є заклад,або підприємство,процеси його проектування,реалізація,наповнення .

Предметом для дослідження є база даних інтернет-магазину комп’ютерної техніки і процеси її проектування та розробки.

Методи,що були використані при розв’язанні задачі – експериментально-теоретичні:порівняння, узагальнення, аналіз,моделювання.

Результатом цього курсового проекту є одна база даних і сайт для її використання.

Ключові слова: БАЗА ДАНИХ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ,СИСТЕМИ УПРАВЛІННЯ БАЗАМИ ДАННИХ, РОЗРОБКА БАЗ ДАНИХ, АНАЛІЗ ПРЕДМЕТНОЇ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ,РЕАЛІЗАЦІЯ БАЗ ДАНИХ ІНТЕРНЕТМАГАЗИНУКОМПЮТЕРНОЇТЕХНІКИ

ЗМІСТ

[ВСТУП 5](#_Toc482023750)

[АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7](#_Toc482023751)

[ПОСТАНОВКА ЗАДАЧІ 8](#_Toc482023752)

[ПРОЕКТУВАННЯ БАЗИ ДАНИХ 9](#_Toc482023753)

[Виявлення основних сутностей предметної області 9](#_Toc482023754)

[Перерахування всіх основних сутностей 9](#_Toc482023755)

[Відношення між сутностями 12](#_Toc482023756)

[Побудова схеми реляційної бази даних 14](#_Toc482023757)

[ОПИС КОНЦЕПТУАЛЬНОЇ МОДЕЛІ БАЗ ДАНИХ 21](#_Toc482023758)

[Короткі характеристики обраних СУБД 21](#_Toc482023759)

[OpenOffice Base 21](#_Toc482023760)

[Microsoft Office Access 22](#_Toc482023761)

[MySQL 23](#_Toc482023762)

[Опис реалізації в кожній з обраних СУБД 24](#_Toc482023763)

[Реалізація в OpenOffice Base 24](#_Toc482023764)

[Реалізація в Microsoft Access 28](#_Toc482023765)

[Реалізація MySQL 32](#_Toc482023766)

[ВИСНОВКИ 38](#_Toc482023767)

[ПОСИЛАННЯ 39](#_Toc482023768)

[Додаток А 40](#_Toc482023769)

[Додаток Б 46](#_Toc482023770)

[Додаток В 48](#_Toc482023771)

# 

# ВСТУП

В наш час розвиток йде впевненими кроками до інформаційного майбутнього.

Впровадження інформаційних технологій зробило можливим значно підвищити ефективність роботи підприємств, організацій, проведення наукових досліджень, підвищити якість роботи в різних галузях освіти, науки, культури. Широке практичне використання технологій баз даних обумовлено значними досягненнями у цій галузі провідними комп'ютерними компаніями світу, потребою суспільства в ефективних засобах зберігання і обробки інформації. Отже, зростає потреба у фахівцях, здатних розробляти і застосовувати

системи баз даних.

Бази даних – це сукупність даних,що організовані відповідно до концепції, яка описує характеристику цих даних і взаємозв’язки між їх елементами. Також вона містить опис цих даних і засоби для їх обробки. Для забезпечення роботи з базою даних використовують системи керування базами даних(надалі СУБД),що являють собою систему для створення, маніпулювання, контролю, керування та використання баз даних. Саме СУБД забезпечує ефективну роботу з базою,обсяг якої є занадто великим для ручної обробки. Зокрема, завдяки цій системі база даних стає автоматизованою.

# ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА

# ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ

# Аналіз інформаційних потреб та визначення предметної області

# дослідження

**“Якщо вашого бізнесу немає в інтернеті, значить вас немає в бізнесі”. ©Білл Гейтс**

З впровадженням інтернет-технологій, інтернет-магазини складають значну конкуренцію реальним магазинам і з кожним роком забирають все більший відсоток замовлень.

І вже зараз не можна заперечити той факт, що в інтернеті проходить значна частина життя сучасної людини. Покупки, спілкування, робота перемістились з офлайн в онлайн, тож і не дивно, що покупки в інтернеті стала більш потужним джерелом торгівлі.

Більшість людей перед тим як здійснити покупку будь-якого товару, переглядають інформацію про товар у мережі Інтернет, порівнюють ціни та характеристики. Якщо ще десять років тому люди користувалися послугами консультантів в магазинах побутової техніки, то тепер, як свідчить внутрішня статистика супермаркетів електроніки, більшість людей приходить у магазин із чітким знанням інформації про відмінності та переваги різних груп товарів.

З цього можна зробити висновок–якщо Вас немає у мережі, то для 60% покупців (і це показники тільки 2013 року) Ваш бізнес просто не існує.

Звичайно бази даних грають невід’ємну частину інтернет-магазину. Завдяки ним можна отримувати наявні данні про товари на складі,замовлення,клієнтів і т.д. Вони пришвидшують і автоматизують додавання, використання, пошук, корегування і видалення даних.

В цьому курсовому проекті досліджується проблема ,котра є актуальної по цей день,майже всі інтернет-магазини мають недоліки. Інтернет-магазин повинен обладати зручною навігацією і найголовніше швидкістю і не тільки швидкістю роботи сайта,але й швидкістю відгука на замовлення,зв’язку з покупцем для узгодження замовлення,та зв’язку з базами даних. Інтернет-магазини з найбільшою швидкістю роботи і зрозумілістю інтерфейса мають більше покупців і складають гарне враження на майбутнє.Це є важливим аспектом,адже згідно статистиці більшість покупців,після вдалої покупки,звернуться на те й же сайт.База даних для інтернет-магазину комп’ютерної техніки потрібна для збереження, редагування, видалення і т.д. даних. Головним завданням створення інтернет-магазину є проектування коректної,без надлишкової інформації бази даних.

Метою курсової роботи є створення бази даних інформаційної системи інтернет-магазину комп’ютерної техніки.

Задачі,що було практично вирішені:

* Вивчення особливостей діяльності інтернет-магазину
* Розробка схеми БД
* Реалізація розробленої схеми в обраній СУБД

Згідно з проведеним аналізом предметної області нам потрібно розробити базу даних,що буде містити таблиці:

**Товари:**

1. Назва
2. Опис
3. Ціна
4. Валюта
5. Фотографія
6. Кількість товару на складі
7. Тип товару

**Клієнтів:**

1. Імя
2. Прізвище
3. Фотографія
4. Адреса
5. Номер телефону

**Особисті дані клієнтів:**

1. E-mail
2. Пароль
3. Права користувача

**Замовлення:**

1. Імя замовника
2. Прізвище замовника
3. Адреса замовника
4. Номер телефону замовника
5. Ціна замовлення
6. Тип доставки
7. Дата замовлення
8. Статус замовлення

**Тіло замовлення:**

1. Назва товару
2. Ціна товару
3. Валюта
4. Кількість
5. Тип товару

Згідно з проведеним аналізом бази даних потрібно розробити веб-сайт для взаємодії з усіма таблицями та властивостями бази даних.

* 1. Обґрунтування вибору засобів реалізації.

**MySQL** - найпопулярніша і поширена СУБД.



**MySQL** - це найпоширеніша повноцінна серверна СУБД. MySQL дуже функціональна, вільно розповсюджується, успішно працює з різними сайтами і веб додатками. Навчитися використанню цієї СУБД досить просто, так як в інтернеті ви легко знайдете велику кількість інформації.

Незважаючи на те, що в ній не реалізований весь SQL функціонал, MySQL пропонує досить багато інструментів для розробки додатків. Так як це серверна СУБД, додатки для доступу до даних, на відміну від SQLite працюють зі службами MySQL.

**Переваги MySQL**

* Простота в роботі - встановити MySQL досить просто.
* Багатий функціонал – MySQL хоча і має не дуже багато функціоналу SQL, але цього цілком достатньо для складних задач.
* Безпека – MySQL має велику кількість функцій які забезпечують безпеку.
* Масштабованість - MySQL легко працює з великими обсягами даних.
* Швидкість - спрощення деяких стандартів дозволяє MySQL значно збільшити продуктивність на обсязі до 500000 записів.
* Безкоштовні відкриті ліцензії.
* Підтримка більшістю хостингових компаній.
* Можливість використання на різних платформах.

**Недоліки MySQL**

* Відомі обмеження - за задумом в MySQL закладені деякі обмеження функціонала.
* Проблеми з надійністю - через деякі способи обробки даних, MySQL (зв'язку, транзакції, аудити) іноді поступається іншим СУБД по надійності.
* Повільна розробка - Хоча MySQL технічно відкрите ПЗ, існують скарги на процес розробки.

**Типи даних MySQL**

TINYINT - дуже малі цілочисельні значення

SMALLINT - малі цілочисельні значення

MEDIUMINT - середні цілочисельні значення

INT або INTEGER - стандартні цілочисельні значення

BIGINT - великі цілочисельні значення

FLOAT - маленькі значення з плаваючою точкою (точність до одного значення після точки). Завжди знакові значення

DOUBLE, BOUBLE PRECISION, REAL - Стандартні значення з плаваючою точкою. завжди знакові

DECIMAL, NUMERIC - розпаковане значення з плаваючою точкою, завжди знакова.

DATE - дата

DATETIME - дата і час в одному значенні

TIMESTAMP - тимчасова відмітка timestamp

TIME - час

YEAR - рік, 2 або 4 числа (4 - за замовчуванням)

CHAR - строкове значення фіксованої довжини, праворуч завжди додаються пропуски перед зазначеної довжини при сортуванні

VARCHAR - строкове значення змінної довжини

ENUM - перерахування

SET – множина

**Коли слід використовувати MySQL**

* Розподілені операції - якщо функціоналу SQLite не вистачає, то варто розглянути MySQL. Так як ця СУБД поєднує в собі просунутий функціонал і вільний доступ до вихідного коду.
* Високий рівень безпеки - система безпеки MySQL включає в себе прості і в той же час гідні способи захисту доступу до даних.
* Веб сайти і веб додатки - більшість сайтів і онлайн додатків спокійно працюють з MySQL незважаючи на деякі обмеження. Будучи легкої в налаштуванні і масштабуванні - MySQL перевірена часом.
* Індивідуальні рішення - якщо ви працюєте з якихось специфічних проектом, MySQL легко зможе вам допомогти завдяки широким можливостям в налаштуванні і функціоналом.

**Коли краще відмовитися від MySQL**

* Відповідність стандартам -Так як MySQL не ставить собі за мету - повністю відповідати стандартам SQL, то ця СУБД в повному обсязі підтримує SQL.
* Многопоточність - хоча деякі движки БД досить легко виконують паралельне читання, паралельні операції читання-запису можуть створити проблеми
* Недолік функціоналу - деякі движки MySQL, наприклад, не підтримують повнотекстовий пошук.

**Висновки з СУБД:**

Підсумувавши усі переваги і недоліки,розібравшись з поставленною в данному курсовому проекті задачею,я вибрав СУБД **PHPMyAdmin**,яка базується на РСУБД MySQL.

**SQLite** - дуже потужна вбудована система управління



Легко вбудовується в додатки база даних. Так як це система базується на файлах, то вона надає досить широкий набір інструментів для роботи з нею, в порівнянні з мережевими СУБД. При роботі з цією СУБД звернення відбуваються безпосередньо до файлів (в ці файлах зберігаються дані), замість портів в мережевих СУБД. Саме тому SQLite дуже швидка, а також потужна завдяки технологіям вбудованих в неї бібліотек.

**Переваги SQLite**

* Файлова структура - вся база даних складається з одного файлу, тому її дуже легко переносити на різні машини
* Стандарти - хоча може здатися, що ця СУБД примітивна, але вона використовує SQL.
* Відмінна при розробці та тестуванні - в процесі розробки додатків часто з'являється необхідність масштабування. SQLite пропонує все що необхідно для цих цілей, так як складається всього з одного файлу і бібліотеки написаної на мові C.

**Недоліки SQLite**

* Відсутність системи користувачів - більші СУБД включають в свій склад системи управління правами доступу користувачів. Зазвичай застосування цієї функції не так критично, так як ця СУБД використовується в невеликих додатках.
* Відсутність можливості збільшення продуктивності - знову, виходячи з проектування, досить складно вичавити щось більш продуктивне з цієї СУБД.

**Коли використовувати SQLite**

* Вбудовані додатки - якщо вам важлива можливість легкого перенесення програми, але не важлива масштабованість. Наприклад однокористувацькі додатки, мобільні додатки або ігри
* Прямий доступ до диска - при необхідності безпосередньо звертатися до диска ви можете виграти при переході на цю СУБД в функціоналі і простоті використання SQL мови
* Тестування - відмінно підійде для більшості додатків, частиною функціоналу яких є тестування бізнес-логіки.

**Коли не варто використовувати SQLite**

* Розраховані на багато користувачів програми: якщо ви працюєте над додатком, доступом до БД в якому будуть одночасно користуватися кілька людей, краще вибрати повнофункціональну СУБД - наприклад, MySQL.
* Додатки, що записують великі обсяги даних: одним з обмежень SQLite є операції записи. Ця СУБД допускає одноразову виконання лише однієї операції записи.

**PostgreSQL** - найбільш просунута СУБД

PostgreSQL - це сама просунута РСУБД, що орієнтується в першу чергу на повну відповідність стандартам і розширюваність. PostgreSQL, або Postgres, намагається повністю відповідати SQL-стандартам ANSI / ISO.

PostgreSQL відрізняється від інших РСУБД тим, що володіє об'єктно-орієнтованим функціоналом.

Хоча ця РСУБД не так популярна, як MySQL, існує

багато сторонніх інструментів і бібліотек для полегшення роботи з PostgreSQL.

**Переваги PostgreSQL**

* Відкрите ПЗ відповідає стандарту SQL - PostgreSQL - безкоштовне ПЗ з відкритим вихідним кодом. Ця СУБД є дуже потужною системою.
* Велике співтовариство - існує досить велика спільнота в якому ви запросто знайдете відповіді на свої питання
* Велика кількість доповнень - незважаючи на величезну кількість вбудованих функцій, існує дуже багато доповнень, що дозволяють розробляти дані для цієї СУБД і управляти ними.
* Розширення - існує можливість розширення функціоналу за рахунок збереження своїх процедур.
* Об'єктно - PostrgreSQL це не тільки реляційна СУБД, але також і об'єктно-орієнтована.

**Недоліки PostgreSQL**

* Продуктивність - при простих операціях читання PostgreSQL може значно уповільнити сервер і бути повільніше своїх конкурентів.
* Популярність - за своєю природою, популярністю ця СУБД похвалитися не може, хоча і є досить велика спільнота.
* Хостинг - в силу названих вище чинників іноді досить складно знайти хостинг з підтримкою цієї СУБД.

**Коли варто використовувати PostgreSQL**

* Цілісність даних: PostgreSQL є найкращим вибором, якщо пріоритет стоїть на надійність і цілісність даних.
* Складні процедури: якщо ваша БД повинна виконувати складні процедури, варто вибрати PostgreSQL в силу її розширюваності.
* Інтеграція: якщо в майбутньому вам доведеться переміщати всю базу на інше рішення, найменше проблем виникне з PostgreSQL.
* Складна структура даних - в порівнянні з іншими відкритими СУБД PostgreSQL надає більше можливостей для створення складних структур даних без необхідності жертовать якими або аспектами.

**Коли не слід використовувати PostgreSQL**

* Швидкість - якщо швидке читання для вас єдиний фактор, то варто придивитися до інших СУБД.
* Просте налаштування - якщо вам не потрібна цілісність даних, відповідність ACID або складні структури даних, то налаштування PostgreSQL може добряче попсувати вам нерви.
* Реплікація - якщо ви не готові витратити час і енергію на те, що міг би з легкістю зробити MySQL, то напевно простіше було б на ньому і залишитися.

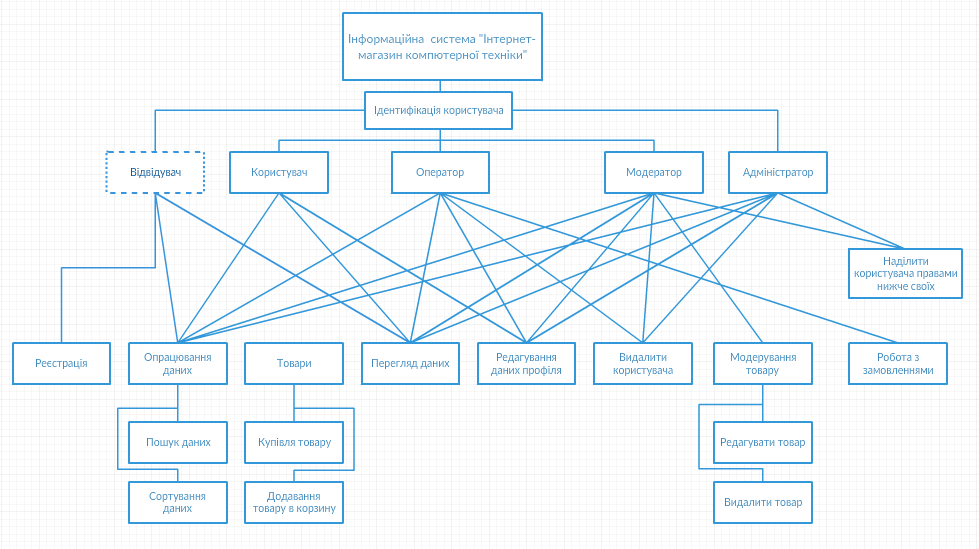
1. ПРОЕКТУВАННЯ БАЗИ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ

РОБОТИ

# ПРОЕКТУВАННЯ БАЗИ ДАНИХ

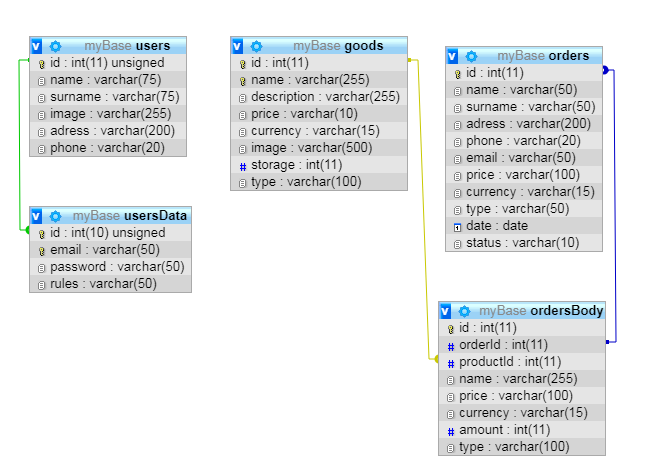
* 1. Проектування структури бази даних за напрямом курсової роботи

Данна схема була розроблена після аналізу архітектури та функцій даного курсового проекту.



**РИС.2.1. Схема ілюстрації роботи системи**

* 1. Проектування структури бази даних за напрямом курсової роботи



**РИС.2.2. Логічну ERD діаграму**

**Товар** – має поля,введені модератором, і мають наступний вид:

1. ID – унікальний номер товару
2. Назва
3. Опис
4. Ціна
5. Валюта
6. URL Фотографії
7. Кількість товару на складі
8. Тип товару

**Користувач** – має поля,що описують зареєстровану користувачем запис і мають наступний вид:

1. ID – унікальній номер користувача
2. Імя
3. Прізвище
4. URL Фотографії
5. Адреса
6. Номер телефону

**Особисті дані користувачів –** має поля,які вводяться користувачем при реєстрації,але вони повинні бути засекречені**:**

1. ID – унікальний номер для зв’язку з таблицею користувачів
2. E-mail
3. Пароль
4. Права користувача

**Замовлення:**

1. ID – універсальний номер замовлення
2. Імя замовника
3. Прізвище замовника
4. Адреса замовника
5. Номер телефону замовника
6. Ціна замовлення
7. Тип доставки
8. Дата замовлення
9. Статус замовлення

**Тіло замовлення:**

1. ID
2. ID замовлення – номер для зв’язку з таблицею замовлень
3. ID продукту – номер для зв’язку з таблицею товарів
4. Назва товару
5. Ціна товару
6. Валюта
7. Кількість
8. Тип товару

### **Відношення між сутностями:**

1. Відношення **1:1** між атрибутом ID y **користувача** і атрибутом ID y **особистих даних користувача**,оскільки один користувач може мати лише одну запис з особистими даними.
2. Відношення **1:M** між атрибутом ID у **замовлення** і атрибутом ID замовлення у **тіла замовлення**. Це відношення означає що у замовлення може бути багато товарів у вигляді стовбців таблиці **тіло замовлення**.
3. Відношення **1:1** між атрибутом ID y **товару** і атрибутом ID продукту у **тілі замовлення**. Через те,що у тілі замовлення повинен бути унікальний ID товару.
4. Відношення **1:1** між атрибутом ID у **замовленні** і атрибутом ID продукту у **тілі замовлення**. Через те,що у одного замовлення можуть бути товари лише з унікальними ID.
   1. Алгоритми обробки даних системи
5. РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ РОБОТИ
   1. Проектування інтерфейсу обробки даних

Під час розробки Бази Даних були створені таблиці:

* Goods
* Orders
* OrdersBody
* Users
* UsersData

Більш детальний перелік таблиць БД наведено в табл. Б.1.

Під час розгляду усіх користувачів інформаційної системи «Інтернет-магазин комп’ютерної техніки» були визначені категорії користувачів:керівництво та покупці.

Найбільші права доступу має керівництво, яке необхідно для роботи інтернет-магазину.Керівники можуть назначати правами (модератора,оператора)користувачів, видаляти користувачів.Модератори мають права на редагування, додавання, видалення товару.Оператори можуть приймати або відхиляти замовлення.

Більш детально перелік об’єктів БД, доступ до яких надано ролі

"Керівник" наведено в табл. Б.2. На перетині рядків і стовпців зазначено дії, які

може виконувати користувач даного типу ролі:

1 – перегляд даних, 2 – редагування, 3 – видалення, 4 – повний доступ.

Матриця доступу для ролі «Покупці»

|  |  |  |  |
| --- | --- | --- | --- |
|  | Товари | Замовлення | Профіль |
| Відвідувач | 1 | 0 | 0 |
| Користувач | 1 | 1 | 2 |

Матриця доступу для ролі «Керівники»

|  |  |  |  |
| --- | --- | --- | --- |
|  | Товари | Замовлення | Профіль |
| Адміністратор | 1 | 1 | 4 |
| Модератор | 4 | 1 | 4 |
| Оператор | 1 | 3 | 2 |

## Опис реалізації в кожній з обраних СУБД

### Реалізація в OpenOffice Base

#### Таблиці і їх структура

Applicants

Поля:

1. Id,тип даних – integer,авто значення,первинний ключ
2. FirstName – varchar,довжина – 50 символів,обов’язкове.
3. LastName – varchar,довжина – 50 символів, обов’язкове
4. FathersName – varchar,довжина – 50 символів,обов’язкове
5. RegDate – timestamp, обов’язкове
6. Picture\_url - varchar,довжина – 50 символів, обов’язкове
7. LocationID – integer,обов’язкове
8. Phone – varchar,довжина – 50 символів, обов’язкове
9. BirthDay – date,обов’язкове
10. SchoolID – integer,обов’язкове
11. PassportID – integer,обов’язкове
12. SchoolCertificationID – integer, обов’язкове
13. Registrar – varchar, довжина – 10 символів
14. ContractID – integer, обов’язкове
15. ExamResult – double
16. ZnoID – integer, обов’язкове
17. Interview – Boolean
18. EnrollmentID – integer
19. EnrollmentType – varchar,довжина – 50 символів

Assertions

Поля:

1. AssertionID,тип даних – integer,автозначення ,первинний ключ
2. ID – integer,обов’язкове
3. assertionDate – timestamp,обов’язкове
4. FacultyID – integer, обов’язкове
5. SpecialtyID – integer, обов’язкове
6. assertionPriority – integer, обов’язкове
7. OriginalDocs – Boolean, обов’язкове
8. DormitoryRequirement – Boolean, обов’язкове

Faculties

Поля:

1. FacultyID,тип даних – integer,автозначення, первинний ключ
2. FacultyName, varchar,довжина – 100 символів,обов’язкове

Specialties

Поля:

1. SpecialtyID,тип даних – integer, автозначення, первинний ключ
2. SpecialtyCode – varchar,довжина – 10 символів,обов’язкове
3. SpecialtyName – varchar, довжина – 50 символів, обов’язкове
4. FacultyID – integer, обов’язкове

ZnoCertification

Поля:

1. ZnoID, тип даних – integer, автозначення, первинний ключ
2. ZnoSN – varchar,довжина – 20 символів,обов’язкове
3. IssueDate – date,обов’язкове

Subjects

Поля:

1. SubjectID, тип даних – integer, автозначення, первинний ключ
2. SubjectName – varchar, довжина – 20 символів,обов’язкове

ZnoSubjectGrades

Поля:

1. ZnoID, тип даних – integer, первинний ключ
2. SubjectID – integer, первинний ключ
3. Grade – double, обов’язкове

Passports

Поля:

1. PassportID, тип даних – integer, первинний ключ
2. PassportSN – varchar,довжина – 10 символів, обов’язкове
3. PassportDate – date, обов’язкове
4. PassportRegion – varchar,довжина – 50 символів, обов’язкове

SchoolCertifications

Поля:

1. SchoolCertificationID, тип даних – integer, первинний ключ
2. SchoolCertificationSN – varchar, довжина – 15 символів,обов’язкове
3. SchoolID – integer,обов’язкове
4. AverageGrade – float,обов’язкове

Schools

Поля:

1. SchoolID,тип даних – integer, первинний ключ
2. SchoolName – varchar,довжина – 100 символів, обов’язкове
3. SchoolLocationID – integer, обов’язкове

Contracts

Поля:

1. ContractID, тип даних – integer, первинний ключ
2. CompanyID – integer,обов’язкове
3. ContractAbout – longvarchar,обов’язкове
4. ContPhone – varchar, довжина – 15 символів,обов’язкове
5. Payment – Boolean,обов’язкове

Companies

Поля:

1. CompanyID, тип даних – integer, первинний ключ
2. CompanyName – varchar, довжина – 50 символів,обов’язкове
3. CompanyLocationID – integer, обов’язкове

Locations

Поля:

1. LocationID, тип даних – integer,первинний ключ
2. CountryID – integer, обов’язкове
3. RegionID – integer, обов’язкове
4. DistrictID – integer, обов’язкове
5. TownID – integer, обов’язкове
6. StreetID – integer, обов’язкове
7. HouseNum – varchar, довжина – 8 символів, обов’язкове
8. ZIP\_code – integer, обов’язкове

Countries

Поля:

1. CountryID, тип даних – integer, первинний ключ
2. CountryName – varchar, довжина – 70 символів,обов’язкове

Regions

Поля:

1. RegionID,тип даних – integer, первинний ключ
2. RegionName – varchar, довжина – 70 символів, обов’язкове
3. CountryID – integer,обов’язкове

Districts

Поля:

1. DistrictID,тип даних – integer, первинний ключ
2. DistrictName – varchar, довжина – 70 символів, обов’язкове
3. RegionID – integer,обов’язкове

Towns

Поля:

1. TownID,тип даних – integer, первинний ключ
2. TownName – varchar, довжина – 70 символів, обов’язкове
3. DistrictID – integer,обов’язкове

Streets

Поля:

1. StreetID,тип даних – integer, первинний ключ
2. StreetName – varchar, довжина – 70 символів, обов’язкове
3. TownID – integer,обов’язкове

#### Зв’язки між таблицями

Всі зв’язки між таблицями і їх атрибутами показані на малюнку нижче

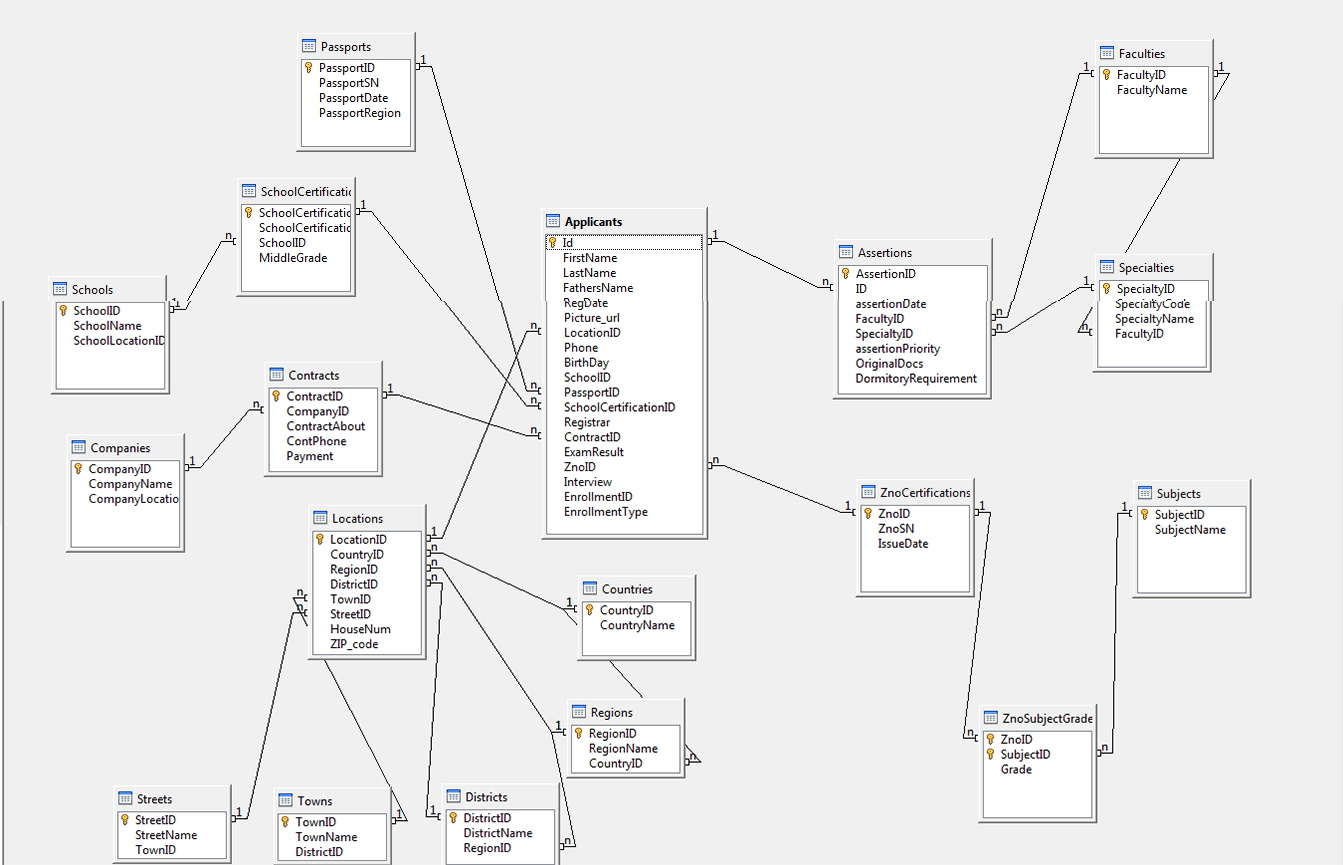


Рисунок 5.6 - Зв’язки між таблицями в реалізації OpenOffice Base

### Реалізація в Microsoft Access

#### Таблиці і їх структура

Applicants

Поля:

1. Id,тип даних – автонумерація, індексовано(без повторень),первинний ключ
2. FirstName – текст,розмір поля - 255, обов’язково.
3. LastName – текст,розмір поля – 255, обов’язково
4. FathersName – текст, розмір поля – 255, обов’язково
5. RegDate – дата й час, обов’язково
6. Photo - вкладення , обов’язково
7. LocationID – число,розмір поля – довге ціле число,обов’язково, індексовано(без повторень)
8. Phone – текст,розмір поля – 255, обов’язково
9. BirthDay – дата й час,обов’язково
10. SchoolID – число,розмір поля – довге ціле число, обов’язково
11. PassportID – число, розмір поля – довге ціле число, обов’язково,індексовано(без повторень)
12. SchoolCertificationID – число, розмір поля – довге ціле число, обов’язково,індексовано(без повторень)
13. Registrar – текст, розмір поля – 255,обов’язково
14. ContractID – число, розмір поля – довге ціле число, ,індексовано(без повторень)
15. ExamResult – число, розмір поля – подвійне значення
16. ZnoID – число, розмір поля – довге ціле число, ,індексовано(без повторень)
17. Interview – так/ні
18. EnrollmentID – число, розмір поля – довге ціле число, індексовано(без повторень)
19. EnrollmentType – текст,розмір поля – довге ціле число

Assertions

Поля:

1. AssertionID,тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. ID –число, розмір – довге ціле число, обов’язково
3. assertionDate – дата і час, обов’язково
4. FacultyID – число, розмір – довге ціле число, обов’язково
5. SpecialtyID – число, розмір – довге ціле число, обов’язково
6. assertionPriority – число,розмір – довге ціле число, обов’язково
7. OriginalDocs – так/ні
8. DormitoryRequirement – так/ні

Faculties

Поля:

1. FacultyID - число, автонумерація,індексовано(без повторень), первинний ключ
2. FacultyName- текст, розмір поля – 255, обов’язково

Specialties

Поля:

1. SpecialtyID,тип даних - автонумерація, індексація(без повторень), первинний ключ
2. SpecialtyCode – текст, розмір поля – 255, обов’язково
3. SpecialtyName – текст, розмір поля – 255, обов’язково
4. FacultyID – число, розмір – довге ціле число, обов’язково

ZnoCertifications

Поля:

1. ZnoID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. ZnoSN – текст,розмір поля – 255, обов’язково
3. IssueDate – дата і час, обов’язково

Subjects

Поля:

1. SubjectID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. SubjectName – текст,розмір поля – 255, обов’язково

ZnoSubjectGrades

Поля:

1. ZnoID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. SubjectID – число, розмір поля – довге ціле число, індексовано(без повторень), первинний ключ
3. Grade – double, обов’язкове

Passports

Поля:

1. PassportID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. PassportSN – текст, розмір поля – 255, обов’язково
3. PassportDate – дата і час, обов’язково
4. PassportRegion – текст, розмір поля – 255, обов’язково

SchoolCertifications

Поля:

1. SchoolCertificationID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. SchoolCertificationSN – текст, розмір поля – 255 символів, обов’язково
3. SchoolID – число, розмір поля – довге ціле число, обов’язково
4. AverageGrade – число, розмір поля – подвійне значення, обов’язково

Schools

Поля:

1. SchoolID,тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. SchoolName – текст, розмір поля – 255, обов’язково
3. SchoolLocationID – число, розмір поля – довге ціле число, індексовано(без повторень), обов’язково

Contracts

Поля:

1. ContractID, тип даних – автонумерація, індексація(без повторень), первинний ключ
2. CompanyID – число, розмір – довге ціле число, обов’язково
3. ContractAbout – примітка, обов’язково
4. ContPhone – текст, розмір поля – 255,обов’язково
5. Payment – так/ні

Companies

Поля:

1. CompanyID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. CompanyName – текст, розмір поля – 255, обов’язково
3. CompanyLocationID – число, розмір поля – довге ціле число, обов’язково

Locations

Поля:

1. LocationID, тип даних – авто нумерація,індексовано(без повторень), первинний ключ
2. CountryID – ціле число, розмір поля – довге ціле число обов’язково
3. RegionID – ціле число, розмір поля – довге ціле число обов’язково
4. DistrictID – ціле число, розмір поля – довге ціле число обов’язково
5. TownID – ціле число, розмір поля – довге ціле число обов’язково
6. StreetID – ціле число, розмір поля – довге ціле число обов’язково
7. HouseNum – varchar, довжина – 8 символів, обов’язкове
8. ZIP\_code – ціле число, обов’язково

Countries

Поля:

1. CountryID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. CountryName – текст, розмір поля - 255, обов’язково

Regions

Поля:

1. RegionID, тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. RegionName – текст, розмір поля - 255, обов’язково
3. CountryID – число, розмір поля – довге ціле число, обов’язково

Districts

Поля:

1. DistrictID,тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. DistrictName – текст, розмір поля – 255, обов’язково
3. RegionID – число, розмір поля – довге ціле число, обов’язково

Towns

Поля:

1. TownID,тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. TownName – текст, розмір поля – 255, обов’язково
3. DistrictID – число, розмір поля –довге ціле число, обов’язково

Streets

Поля:

1. StreetID,тип даних – автонумерація, індексовано(без повторень), первинний ключ
2. StreetName – текст, розмір поля - 255, обов’язково
3. TownID – текст, розмір поля – довге ціле число, обов’язково

#### Зв’язки між таблицями

Всі зв’язки між таблицями і їх атрибутами показані на малюнку нижче

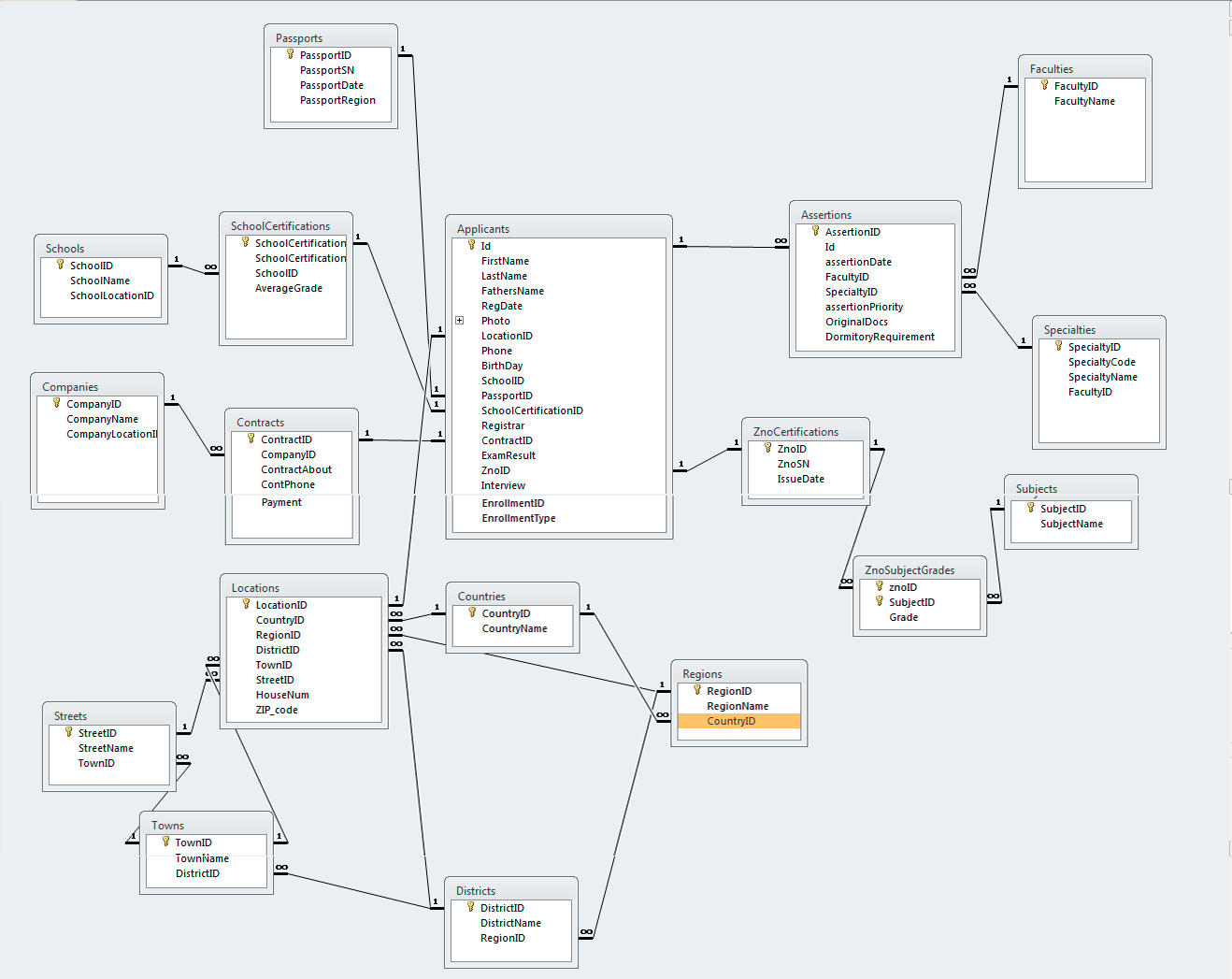


Рисунок 5.7 – Зв’язки між таблицями в реалізації Microsoft Access

### Реалізація MySQL

#### Таблиці і їх структура

Applicants

Поля:

1. Id,тип даних – int ,autoincrement, not null
2. FirstName – varchar(100) , not null
3. LastName – varchar(100) , not null
4. FathersName – varchar(100) , not null
5. RegDate – timestamp,not null
6. Photo\_url - varchar(100), not null
7. LocationID – int, not null
8. Phone – varchar(15), not null
9. BirthDay - date, not null
10. SchoolID – int, not null
11. PassportID – int, not null
12. SchoolCertificationID – int, not null
13. Registrar – varchar(300), not null
14. ContractID – int, null
15. ExamResult – double(4,3), null
16. ZnoID – int, not null
17. Interview – bool, not null
18. EnrollmentID – int, null
19. EnrollmentType – varchar(50), null

Unique index – Applicants\_znoID(znoID)

Unique index – Applicants\_PassportID(PassportID)

Unique index – Applicants\_SchoolCertificationID(SchoolCertificationID)

Unique index – Applicants\_ContractID(ContractID)

Unique index – Applicants\_EnrollmentID(EnrollementID)

Unique index – Applicants\_LocationID(LocationID)

Обмеження Applicants\_pk – первинний ключ(Id)

Assertions

Поля:

1. AssertionID – int, not null, auto\_increment
2. ID – int, not null
3. assertionDate – timestamp, not null
4. FacultyID – int, not null
5. SpecialtyID – int, not null
6. assertionPriority – int, not null
7. OriginalDocs – bool, not null
8. DormitoryRequirement – так/ні

Обмеження Assertions\_pk – первинний ключ(AssertionID)

Faculties

Поля:

1. FacultyID – int, not null, auto\_increment
2. FacultyName - varchar(100), not null

Unique index Faculties\_FacultyName(FacultyName)

Обмеження Faculties\_pk – первинний ключ(FacultyName)

Specialties

Поля:

1. SpecialtyID – int, not null, auto\_increment
2. SpecialtyCode – varchar(10), not null
3. SpecialtyName – varchar(200), not null
4. FacultyID – int, not null

Unique index Specialties\_SpecialtyCode(SpecialtyCode)

Unique index Specialties\_SpecialtyName(SpecialtyName),

Обмеження Specialties\_pk – первинний ключ(SpecialtyID)

ZnoCertifications

Поля:

1. ZnoID – int, not null, auto\_increment
2. ZnoSN – varchar(20), not null
3. IssueDate – timestamp, not null

Unique index ZnoCertifications\_ZnoSN(ZnoSN)

Обмеження ZnoCertification\_pk – первинний ключ(ZnoID)

Subjects

Поля:

1. SubjectID – int, not null, auto\_increment
2. SubjectName – varchar(50), not null

Unique index Subjects\_SubjectName (SubjectName)

Обмеження Subjects\_pk – первинний ключ(SubjectID)

ZnoSubjectGrades

Поля:

1. ZnoID – int,not null, auto\_increment
2. SubjectID - int, not null
3. Grade – float, not null

Обмеження ZnoCertification\_pk – первинний ключ(znoID, SubjectID)

Passports

Поля:

1. PassportID – int, not null, auto\_increment
2. PassportSN – varchar(10), not null
3. PassportDate – date, not null
4. PassportRegion – varchar(150), not null

Unique index Passports\_PassportSN (PassportSN)

Обмеження Passports\_pk – первинний ключ(PassportID)

SchoolCertifications

Поля:

1. SchoolCertificationID – int, not null, auto\_increment
2. SchoolCertificationSN – varchar(15), not null
3. SchoolID – int, not null
4. AverageGrade – float, not null

Unique index SchoolCertifications\_SchoolCertificationID(SchoolCertificationSN)

Обмеження SchoolCertifications\_pk – первинний ключ(SchoolCertificationID)

Schools

Поля:

1. SchoolID – int, not null, auto\_increment
2. SchoolName – varchar(100), not null
3. SchoolLocationID – int, not null

Unique index Schools\_SchoolLocationID(SchoolLocationID)

Обмеження Schools\_pk – первинний ключ (SchoolID)

Contracts

Поля:

1. ContractID – int, not null, auto\_increment
2. CompanyID – int, not null
3. ContractAbout – text, not null
4. ContPhone – varchar(15), not null
5. Payment – bool, not null

Обмеження Contracts\_pk – первинний ключ(ContractID)

Companies

Поля:

1. CompanyID – int, not null
2. CompanyName – varchar(50), not null
3. CompanyLocationID – int, not null

Unique index Companies\_CompanyName(CompanyName)

Unique index Companies\_CompanyLocationID (CompanyLocationID)

Обмеження Companies\_pk – первинний ключ(CompanyID)

Locations

Поля:

1. LocationID – int,not null, auto\_increment
2. CountryID – int, not null
3. RegionID – int, not null
4. DistrictID – int, not null
5. TownID – int, not null
6. StreetID – int, not null
7. HouseNum – varchar(8), not null
8. ZIP\_code – int, not null

Обмеження Locations\_pk – первинний ключ(LocationID)

Countries

Поля:

1. CountryID – int, not null, auto\_increment
2. CountryName – varchar(100), not null

Unique index Countries\_CountryName(CountryName)

Обмеження Countries\_pk – первинний ключ(CountryID)

Regions

Поля:

1. RegionID – int, not null, auto-\_increment
2. RegionName – varchar(150), not null
3. CountryID – int, not null

Unique index Regions\_RegionName(RegionName)

Обмеження Regions\_pk – первинний ключ(RegionID)

Districts

Поля:

1. DistrictID – int, not null, auto\_increment
2. DistrictName – varchar(150), not null
3. RegionID – int,not null

Unique index Districts \_DistrictName(DistrictName)

Обмеження Districts\_pk – первинний ключ(DistrictID)

Towns

Поля:

1. TownID – int, not null, auto\_increment
2. TownName – varchar(150), not null
3. DistrictID – int, not null

Unique index Towns \_TownName(TownName)

Обмеження Towns\_pk – первинний ключ(TownID)

Streets

Поля:

1. StreetID – int,not null, auto\_increment
2. StreetName – varchar(200), not null
3. TownID – int, not null

Unique index Streets \_StreetName(StreetName)

Обмеження Streets\_pk – первинний ключ(StreetID)

#### Зв’язки між таблицями

Всі зв’язки між таблицями і їх атрибутами показані на малюнку нижче

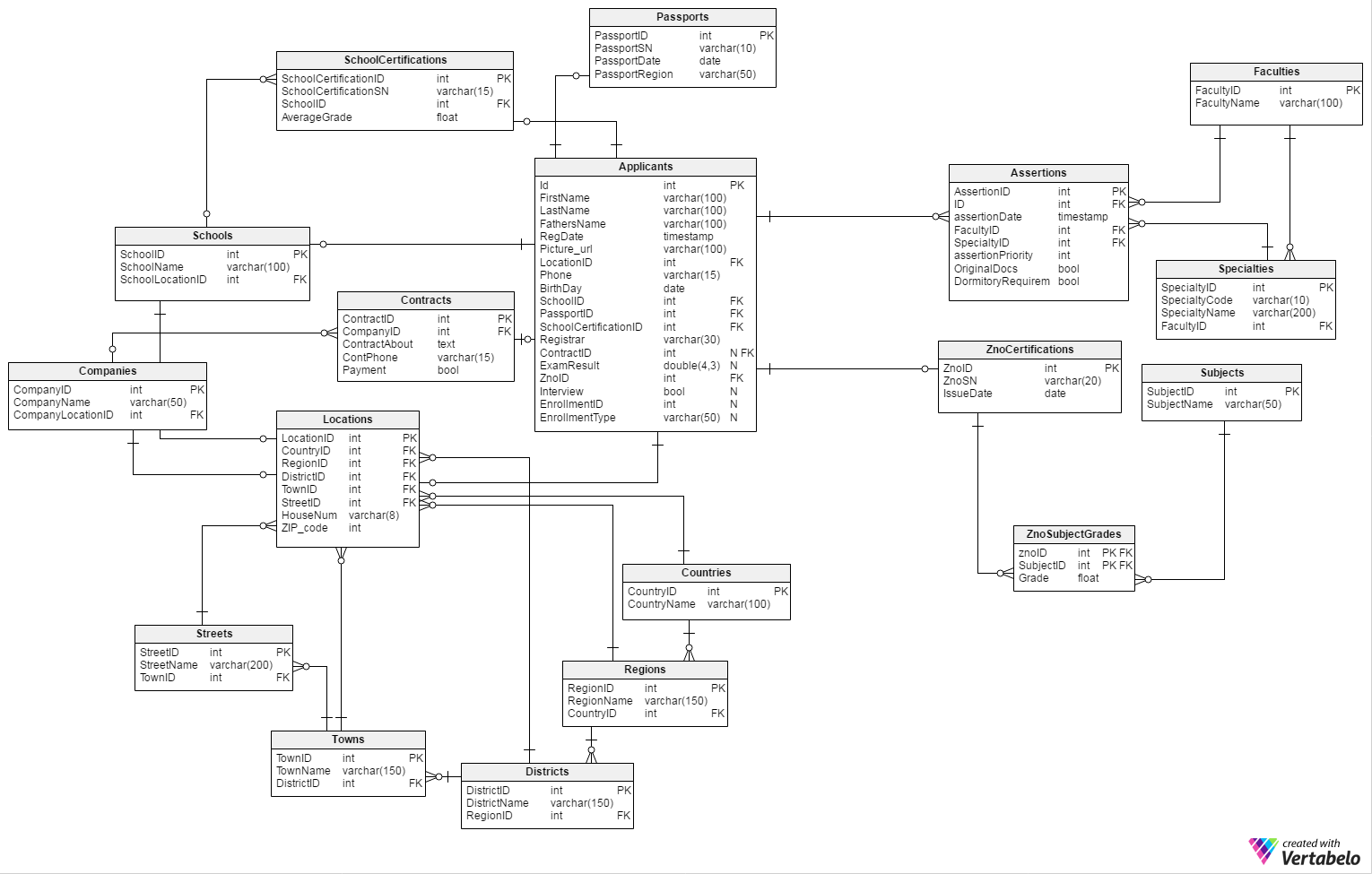


Рисунок 5.8 – Зв’язки між таблицями в MySQL

Повний тексти скриптів на створення таблиці, її видалення, а також запит на додавання нових даних присутні в додатках А,Б,В відповідно.

# ВИСНОВКИ

Виконаний проект не може повноцінно застосовуватися в діяльності вступної комісії університету, оскільки він слугує лише вступників,які будуть здобувачами ступеню бакалавра. В ньому не було враховано магістрів і спеціалістів, а також повністю такі форми навчання як заочна і вечірня. Також я б запропонував би додати в базу даних таблиці, що будуть слугувати для обліку працівників вступної компанії. Для повноцінного використання, база даних має бути набагато розширена, але деякі ідеї, що використані в ній(наприклад ідея багатьох таблиць,для даних про місце розташування, для усунення дуплікації даних) можуть бути використані і в справжніх проектах.

В курсовому проекті на тему «інформаційна система вступної комісії університету» проаналізовано предметну область діяльності університету в сфері вступної компанії. Згідно проведеного аналізу була виконана постановка задачі. Вона складалася з розробки загальної моделі бази даних для вступної комісії університету. Після цього було проведено проектування бази даних, яке складалося з виявлення сутностей предметної області і також встановлення зв’язків між ними. Наступним етапом стало доведення того, що база даних вже знаходиться в третій нормальній формі. Для цього було розглянуто кожну сутність окремо і для кожної сутності було доведено що вона знаходиться спочатку в першій нормальній формі, потім в другій нормальній формі, а потім і в третій нормальній формі. Після цього постало завдання реалізувати модель в обраних СУБД, а саме в OpenOffice Base, Microsoft Access, MySQL. Результатом виконання цього етапу курсової стали реалізовані бази даних в популярних СУБД. Для створення і наповнення баз даних в деяких СУБД було використано SQL скрипти. Під час виконання курсової було вивчено мову структурованих запитів SQL, було здійснено ознайомлення з деякими популярними СУБД, а також загальні відомості про бази даних, в особливості про реляційні бази даних. Було отримано вміння проектування баз даних а також процес нормалізації баз даних, під час якого я дізнався що таке нормальні форми баз даних. Всі завдання,що були заплановані в постановці задачі курсової роботи – виконано, отже, курсову роботу можна вважати виконаною.

# ПОСИЛАННЯ

# Додаток А

Скрипт для створення баз даних

**CREATE** **TABLE** Applicants **(**

Id **int** **NOT** **NULL** AUTO\_INCREMENT**,**

FirstName **varchar(**100**)** **NOT** **NULL,**

LastName **varchar(**100**)** **NOT** **NULL,**

FathersName **varchar(**100**)** **NOT** **NULL,**

RegDate **timestamp** **NOT** **NULL,**

Picture\_url **varchar(**100**)** **NOT** **NULL,**

LocationID **int** **NOT** **NULL,**

Phone **varchar(**15**)** **NOT** **NULL,**

BirthDay **date** **NOT** **NULL,**

SchoolID **int** **NOT** **NULL,**

PassportID **int** **NOT** **NULL,**

SchoolCertificationID **int** **NOT** **NULL,**

Registrar **varchar(**30**)** **NOT** **NULL,**

ContractID **int** **NULL,**

ExamResult **double(**4**,**3**)** **NULL,**

ZnoID **int** **NOT** **NULL,**

Interview bool **NULL,**

EnrollmentID **int** **NULL,**

EnrollmentType **varchar(**50**)** **NULL,**

**UNIQUE** **INDEX** Applicants\_znoID **(**ZnoID**),**

**UNIQUE** **INDEX** Applicants\_PassportID **(**PassportID**),**

**UNIQUE** **INDEX** Applicants\_SchoolCertificationID **(**SchoolCertificationID**),**

**UNIQUE** **INDEX** Applicants\_ContractID **(**ContractID**),**

**UNIQUE** **INDEX** Applicants\_EnrollmentID **(**EnrollmentID**),**

**UNIQUE** **INDEX** Applicants\_LocationID **(**LocationID**),**

**CONSTRAINT** Applicants\_pk **PRIMARY** **KEY** **(**Id**)**

**);**

-- Table: Assertions

**CREATE** **TABLE** Assertions **(**

AssertionID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

ID **int** **NOT** **NULL,**

assertionDate **timestamp** **NOT** **NULL,**

FacultyID **int** **NOT** **NULL,**

SpecialtyID **int** **NOT** **NULL,**

assertionPriority **int** **NOT** **NULL,**

OriginalDocs bool **NOT** **NULL,**

DormitoryRequirement bool **NOT** **NULL,**

**CONSTRAINT** Assertions\_pk **PRIMARY** **KEY** **(**AssertionID**)**

**);**

-- Table: Companies

**CREATE** **TABLE** Companies **(**

CompanyID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

CompanyName **varchar(**50**)** **NOT** **NULL,**

CompanyLocationID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Companies\_CompanyLocationID **(**CompanyLocationID**),**

**UNIQUE** **INDEX** Companies\_CompanyName **(**CompanyName**),**

**CONSTRAINT** Companies\_pk **PRIMARY** **KEY** **(**CompanyID**)**

**);**

-- Table: Contracts

**CREATE** **TABLE** Contracts **(**

ContractID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

CompanyID **int** **NOT** **NULL,**

ContractAbout text **NOT** **NULL,**

ContPhone **varchar(**15**)** **NOT** **NULL,**

Payment bool **NOT** **NULL,**

**CONSTRAINT** Contracts\_pk **PRIMARY** **KEY** **(**ContractID**)**

**);**

-- Table: Countries

**CREATE** **TABLE** Countries **(**

CountryID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

CountryName **varchar(**100**)** **NOT** **NULL,**

**UNIQUE** **INDEX** Countries\_CountryName **(**CountryName**),**

**CONSTRAINT** Countries\_pk **PRIMARY** **KEY** **(**CountryID**)**

**);**

-- Table: Districts

**CREATE** **TABLE** Districts **(**

DistrictID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

DistrictName **varchar(**150**)** **NOT** **NULL,**

RegionID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Districts\_DistrictName **(**DistrictName**),**

**CONSTRAINT** Districts\_pk **PRIMARY** **KEY** **(**DistrictID**)**

**);**

-- Table: Faculties

**CREATE** **TABLE** Faculties **(**

FacultyID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

FacultyName **varchar(**100**)** **NOT** **NULL,**

**UNIQUE** **INDEX** Faculties\_FacultyName **(**FacultyName**),**

**CONSTRAINT** Faculties\_pk **PRIMARY** **KEY** **(**FacultyID**)**

**);**

-- Table: Locations

**CREATE** **TABLE** Locations **(**

LocationID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

CountryID **int** **NOT** **NULL,**

RegionID **int** **NOT** **NULL,**

DistrictID **int** **NOT** **NULL,**

TownID **int** **NOT** **NULL,**

StreetID **int** **NOT** **NULL,**

HouseNum **varchar(**8**)** **NOT** **NULL,**

ZIP\_code **int** **NOT** **NULL,**

**CONSTRAINT** Locations\_pk **PRIMARY** **KEY** **(**LocationID**)**

**);**

-- Table: Passports

**CREATE** **TABLE** Passports **(**

PassportID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

PassportSN **varchar(**10**)** **NOT** **NULL,**

PassportDate **date** **NOT** **NULL,**

PassportRegion **varchar(**50**)** **NOT** **NULL,**

**UNIQUE** **INDEX** Passports\_PassportSN **(**PassportSN**),**

**CONSTRAINT** Passports\_pk **PRIMARY** **KEY** **(**PassportID**)**

**);**

-- Table: Regions

**CREATE** **TABLE** Regions **(**

RegionID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

RegionName **varchar(**150**)** **NOT** **NULL,**

CountryID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Regions\_RegionName **(**RegionName**),**

**CONSTRAINT** Regions\_pk **PRIMARY** **KEY** **(**RegionID**)**

**);**

-- Table: SchoolCertifications

**CREATE** **TABLE** SchoolCertifications **(**

SchoolCertificationID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

SchoolCertificationSN **varchar(**15**)** **NOT** **NULL,**

SchoolID **int** **NOT** **NULL,**

AverageGrade **float** **NOT** **NULL,**

**UNIQUE** **INDEX** SchoolCertifications\_SchoolSertificationID **(**SchoolCertificationSN**),**

**CONSTRAINT** SchoolCertifications\_pk **PRIMARY** **KEY** **(**SchoolCertificationID**)**

**);**

-- Table: Schools

**CREATE** **TABLE** Schools **(**

SchoolID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

SchoolName **varchar(**100**)** **NOT** **NULL,**

SchoolLocationID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Schools\_LocationID **(**SchoolLocationID**),**

**CONSTRAINT** Schools\_pk **PRIMARY** **KEY** **(**SchoolID**)**

**);**

-- Table: Specialties

**CREATE** **TABLE** Specialties **(**

SpecialtyID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

SpecialtyCode **varchar(**10**)** **NOT** **NULL,**

SpecialtyName **varchar(**200**)** **NOT** **NULL,**

FacultyID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Specialties\_SpecialtyCode **(**SpecialtyCode**),**

**UNIQUE** **INDEX** Specialties\_SpecialtyName **(**SpecialtyName**),**

**CONSTRAINT** Specialties\_pk **PRIMARY** **KEY** **(**SpecialtyID**)**

**);**

-- Table: Streets

**CREATE** **TABLE** Streets **(**

StreetID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

StreetName **varchar(**200**)** **NOT** **NULL,**

TownID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Streets\_StreetName **(**StreetName**),**

**CONSTRAINT** Streets\_pk **PRIMARY** **KEY** **(**StreetID**)**

**);**

-- Table: Subjects

**CREATE** **TABLE** Subjects **(**

SubjectID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

SubjectName **varchar(**50**)** **NOT** **NULL,**

**UNIQUE** **INDEX** Subjects\_SubjectName **(**SubjectName**),**

**CONSTRAINT** Subjects\_pk **PRIMARY** **KEY** **(**SubjectID**)**

**);**

-- Table: Towns

**CREATE** **TABLE** Towns **(**

TownID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

TownName **varchar(**150**)** **NOT** **NULL,**

DistrictID **int** **NOT** **NULL,**

**UNIQUE** **INDEX** Towns\_TownName **(**TownName**),**

**CONSTRAINT** Towns\_pk **PRIMARY** **KEY** **(**TownID**)**

**);**

-- Table: ZnoCertifications

**CREATE** **TABLE** ZnoCertifications **(**

ZnoID **int** **NOT** **NULL** AUTO\_INCREMENT**,**

ZnoSN **varchar(**20**)** **NOT** **NULL,**

IssueDate **date** **NOT** **NULL,**

**UNIQUE** **INDEX** ZnoCertifications\_ZnoSN **(**ZnoSN**),**

**CONSTRAINT** ZnoCertifications\_pk **PRIMARY** **KEY** **(**ZnoID**)**

**);**

-- Table: ZnoSubjectGrades

**CREATE** **TABLE** ZnoSubjectGrades **(**

znoID **int** **NOT** **NULL,**

SubjectID **int** **NOT** **NULL,**

Grade **float** **NOT** **NULL,**

**CONSTRAINT** ZnoSubjectGrades\_pk **PRIMARY** **KEY** **(**znoID**,**SubjectID**)**

**);**

-- foreign keys

-- Reference: Applicants\_Contracts (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** Applicants\_Contracts **FOREIGN** **KEY** Applicants\_Contracts **(**ContractID**)**

**REFERENCES** Contracts **(**ContractID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Applicants\_Locations (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** Applicants\_Locations **FOREIGN** **KEY** Applicants\_Locations **(**LocationID**)**

**REFERENCES** Locations **(**LocationID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Applicants\_Passports (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** Applicants\_Passports **FOREIGN** **KEY** Applicants\_Passports **(**PassportID**)**

**REFERENCES** Passports **(**PassportID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Applicants\_Schools (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** Applicants\_Schools **FOREIGN** **KEY** Applicants\_Schools **(**SchoolID**)**

**REFERENCES** Schools **(**SchoolID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Applicants\_ZnoCertifications (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** Applicants\_ZnoCertifications **FOREIGN** **KEY** Applicants\_ZnoCertifications **(**ZnoID**)**

**REFERENCES** ZnoCertifications **(**ZnoID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Assertions\_Applicants (table: Assertions)

**ALTER** **TABLE** Assertions **ADD** **CONSTRAINT** Assertions\_Applicants **FOREIGN** **KEY** Assertions\_Applicants **(**ID**)**

**REFERENCES** Applicants **(**Id**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

-- Reference: Assertions\_Faculties (table: Assertions)

**ALTER** **TABLE** Assertions **ADD** **CONSTRAINT** Assertions\_Faculties **FOREIGN** **KEY** Assertions\_Faculties **(**FacultyID**)**

**REFERENCES** Faculties **(**FacultyID**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

-- Reference: Assertions\_Specialties (table: Assertions)

**ALTER** **TABLE** Assertions **ADD** **CONSTRAINT** Assertions\_Specialties **FOREIGN** **KEY** Assertions\_Specialties **(**SpecialtyID**)**

**REFERENCES** Specialties **(**SpecialtyID**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

-- Reference: Contracts\_Companies (table: Contracts)

**ALTER** **TABLE** Contracts **ADD** **CONSTRAINT** Contracts\_Companies **FOREIGN** **KEY** Contracts\_Companies **(**CompanyID**)**

**REFERENCES** Companies **(**CompanyID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Districts\_Regions (table: Districts)

**ALTER** **TABLE** Districts **ADD** **CONSTRAINT** Districts\_Regions **FOREIGN** **KEY** Districts\_Regions **(**RegionID**)**

**REFERENCES** Regions **(**RegionID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Companies (table: Companies)

**ALTER** **TABLE** Companies **ADD** **CONSTRAINT** Locations\_Companies **FOREIGN** **KEY** Locations\_Companies **(**CompanyLocationID**)**

**REFERENCES** Locations **(**LocationID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Countries (table: Locations)

**ALTER** **TABLE** Locations **ADD** **CONSTRAINT** Locations\_Countries **FOREIGN** **KEY** Locations\_Countries **(**CountryID**)**

**REFERENCES** Countries **(**CountryID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Districts (table: Locations)

**ALTER** **TABLE** Locations **ADD** **CONSTRAINT** Locations\_Districts **FOREIGN** **KEY** Locations\_Districts **(**DistrictID**)**

**REFERENCES** Districts **(**DistrictID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Regions (table: Locations)

**ALTER** **TABLE** Locations **ADD** **CONSTRAINT** Locations\_Regions **FOREIGN** **KEY** Locations\_Regions **(**RegionID**)**

**REFERENCES** Regions **(**RegionID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Streets (table: Locations)

**ALTER** **TABLE** Locations **ADD** **CONSTRAINT** Locations\_Streets **FOREIGN** **KEY** Locations\_Streets **(**StreetID**)**

**REFERENCES** Streets **(**StreetID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Locations\_Towns (table: Locations)

**ALTER** **TABLE** Locations **ADD** **CONSTRAINT** Locations\_Towns **FOREIGN** **KEY** Locations\_Towns **(**TownID**)**

**REFERENCES** Towns **(**TownID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Regions\_Countries (table: Regions)

**ALTER** **TABLE** Regions **ADD** **CONSTRAINT** Regions\_Countries **FOREIGN** **KEY** Regions\_Countries **(**CountryID**)**

**REFERENCES** Countries **(**CountryID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: SchoolCertifications\_Applicants (table: Applicants)

**ALTER** **TABLE** Applicants **ADD** **CONSTRAINT** SchoolCertifications\_Applicants **FOREIGN** **KEY** SchoolCertifications\_Applicants **(**SchoolCertificationID**)**

**REFERENCES** SchoolCertifications **(**SchoolCertificationID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: SchoolCertifications\_Schools (table: SchoolCertifications)

**ALTER** **TABLE** SchoolCertifications **ADD** **CONSTRAINT** SchoolCertifications\_Schools **FOREIGN** **KEY** SchoolCertifications\_Schools **(**SchoolID**)**

**REFERENCES** Schools **(**SchoolID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Schools\_Locations (table: Schools)

**ALTER** **TABLE** Schools **ADD** **CONSTRAINT** Schools\_Locations **FOREIGN** **KEY** Schools\_Locations **(**SchoolLocationID**)**

**REFERENCES** Locations **(**LocationID**)**

**ON** **DELETE** **SET** **NULL**

**ON** **UPDATE** **CASCADE;**

-- Reference: Specialties\_Faculties (table: Specialties)

**ALTER** **TABLE** Specialties **ADD** **CONSTRAINT** Specialties\_Faculties **FOREIGN** **KEY** Specialties\_Faculties **(**FacultyID**)**

**REFERENCES** Faculties **(**FacultyID**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

-- Reference: Streets\_Towns (table: Streets)

**ALTER** **TABLE** Streets **ADD** **CONSTRAINT** Streets\_Towns **FOREIGN** **KEY** Streets\_Towns **(**TownID**)**

**REFERENCES** Towns **(**TownID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: Towns\_Districts (table: Towns)

**ALTER** **TABLE** Towns **ADD** **CONSTRAINT** Towns\_Districts **FOREIGN** **KEY** Towns\_Districts **(**DistrictID**)**

**REFERENCES** Districts **(**DistrictID**)**

**ON** **DELETE** **RESTRICT**

**ON** **UPDATE** **CASCADE;**

-- Reference: ZnoSubjectGrades\_Subjects (table: ZnoSubjectGrades)

**ALTER** **TABLE** ZnoSubjectGrades **ADD** **CONSTRAINT** ZnoSubjectGrades\_Subjects **FOREIGN** **KEY** ZnoSubjectGrades\_Subjects **(**SubjectID**)**

**REFERENCES** Subjects **(**SubjectID**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

-- Reference: ZnoSubjectGrades\_ZnoCertifications (table: ZnoSubjectGrades)

**ALTER** **TABLE** ZnoSubjectGrades **ADD** **CONSTRAINT** ZnoSubjectGrades\_ZnoCertifications **FOREIGN** **KEY** ZnoSubjectGrades\_ZnoCertifications **(**znoID**)**

**REFERENCES** ZnoCertifications **(**ZnoID**)**

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE;**

# Додаток Б

Скрипт для видалення бази даних

-- foreign keys

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** Applicants\_Contracts**;**

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** Applicants\_Locations**;**

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** Applicants\_Passports**;**

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** Applicants\_Schools**;**

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** Applicants\_ZnoCertifications**;**

**ALTER** **TABLE** Assertions

**DROP** **FOREIGN** **KEY** Assertions\_Applicants**;**

**ALTER** **TABLE** Assertions

**DROP** **FOREIGN** **KEY** Assertions\_Faculties**;**

**ALTER** **TABLE** Assertions

**DROP** **FOREIGN** **KEY** Assertions\_Specialties**;**

**ALTER** **TABLE** Contracts

**DROP** **FOREIGN** **KEY** Contracts\_Companies**;**

**ALTER** **TABLE** Districts

**DROP** **FOREIGN** **KEY** Districts\_Regions**;**

**ALTER** **TABLE** Companies

**DROP** **FOREIGN** **KEY** Locations\_Companies**;**

**ALTER** **TABLE** Locations

**DROP** **FOREIGN** **KEY** Locations\_Countries**;**

**ALTER** **TABLE** Locations

**DROP** **FOREIGN** **KEY** Locations\_Districts**;**

**ALTER** **TABLE** Locations

**DROP** **FOREIGN** **KEY** Locations\_Regions**;**

**ALTER** **TABLE** Locations

**DROP** **FOREIGN** **KEY** Locations\_Streets**;**

**ALTER** **TABLE** Locations

**DROP** **FOREIGN** **KEY** Locations\_Towns**;**

**ALTER** **TABLE** Regions

**DROP** **FOREIGN** **KEY** Regions\_Countries**;**

**ALTER** **TABLE** Applicants

**DROP** **FOREIGN** **KEY** SchoolCertifications\_Applicants**;**

**ALTER** **TABLE** SchoolCertifications

**DROP** **FOREIGN** **KEY** SchoolCertifications\_Schools**;**

**ALTER** **TABLE** Schools

**DROP** **FOREIGN** **KEY** Schools\_Locations**;**

**ALTER** **TABLE** Specialties

**DROP** **FOREIGN** **KEY** Specialties\_Faculties**;**

**ALTER** **TABLE** Streets

**DROP** **FOREIGN** **KEY** Streets\_Towns**;**

**ALTER** **TABLE** Towns

**DROP** **FOREIGN** **KEY** Towns\_Districts**;**

**ALTER** **TABLE** ZnoSubjectGrades

**DROP** **FOREIGN** **KEY** ZnoSubjectGrades\_Subjects**;**

**ALTER** **TABLE** ZnoSubjectGrades

**DROP** **FOREIGN** **KEY** ZnoSubjectGrades\_ZnoCertifications**;**

-- tables

**DROP** **TABLE** Applicants**;**

**DROP** **TABLE** Assertions**;**

**DROP** **TABLE** Companies**;**

**DROP** **TABLE** Contracts**;**

**DROP** **TABLE** Countries**;**

**DROP** **TABLE** Districts**;**

**DROP** **TABLE** Faculties**;**

**DROP** **TABLE** Locations**;**

**DROP** **TABLE** Passports**;**

**DROP** **TABLE** Regions**;**

**DROP** **TABLE** SchoolCertifications**;**

**DROP** **TABLE** Schools**;**

**DROP** **TABLE** Specialties**;**

**DROP** **TABLE** Streets**;**

**DROP** **TABLE** Subjects**;**

**DROP** **TABLE** Towns**;**

**DROP** **TABLE** ZnoCertifications**;**

**DROP** **TABLE** ZnoSubjectGrades**;**

# Додаток В

Скрипт для додавання нової таблиці

**INSERT** **INTO** Countries **VALUES(NULL,** 'Україна'**);**

**SELECT** @last\_country **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Regions **VALUES(NULL,**'Житомирська область'**,**@last\_country**);**

**SELECT** @last\_region **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Districts **VALUES(NULL,** 'Малинський район'**,**@last\_region**);**

**SELECT** @last\_district **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Towns **VALUES(NULL,** 'Малин'**,** @last\_district**);**

**SELECT** @last\_town **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Streets **VALUES(NULL,** 'вулиця Мирутенка'**,**@last\_town**);**

**SELECT** @last\_street **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Locations **VALUES(NULL,**@last\_country**,**@last\_region**,**@last\_district**,**@last\_town**,**@last\_street**,**'36/1'**,**11601**);**

**SELECT** @last\_applicant\_location **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Streets **VALUES(NULL,** 'вулиця Кримського'**,**@last\_town**);**

**SELECT** @last\_sstreet **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Locations **VALUES(NULL,**@last\_country**,**@last\_region**,**@last\_district**,**@last\_town**,**@last\_sstreet**,**'5'**,**11601**);**

**SELECT** @last\_school\_loc **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Schools **VALUES(NULL,**'Малинська загальноосвітня школа №2 1-3 ступенів'**,**@last\_school\_loc**);**

**SELECT** @last\_school **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** SchoolCertifications **VALUES(NULL,**'ТН12345'**,**1**,**10.90**);**

**SELECT** @last\_school\_certification **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Passports **VALUES(NULL,**'ВН605661'**,**'2013-10-26'**,**'Малинський РВ УДМС'**);**

**SELECT** @last\_passport **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Faculties **VALUES(NULL,**'Факультет інформатики'**);**

**SELECT** @last\_faculty **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Specialties **VALUES(NULL,**'6.050103'**,**'Програмна інженерія'**,**@last\_faculty**);**

**SELECT** @last\_specialty **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Subjects **VALUES(NULL,**'Українська мова'**),(NULL,**'Фізика'**),(NULL,**'Математика'**);**

**INSERT** **INTO** ZnoCertifications **VALUES(NULL,**'ВС123152'**,**'2015-01-07'**);**

**SELECT** @last\_zno\_certification **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** ZnoSubjectGrades **VALUES(**@last\_zno\_certification**,**1**,**190**),(**@last\_zno\_certification**,**2**,**183**),(**@last\_zno\_certification**,**3**,**196**);**

**INSERT** **INTO** Streets **VALUES(NULL,**'вулиця Гагаріна'**,**@last\_town**);**

**SELECT** @last\_cstreet **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Locations **VALUES** **(NULL,**@last\_country**,**@last\_region**,**@last\_district**,**@last\_town**,**@last\_cstreet**,**'45a'**,**11603**);**

**SELECT** @last\_company\_location **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Companies **VALUES(NULL,**'ЗАТ Вайдманн'**,**@last\_company\_location**);**

**SELECT** @last\_company **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Contracts **VALUES** **(NULL,**@last\_company**,**'Контракт для Хоменко Андрія від компанії Вайдманн'**,**'+380413334256'**,TRUE);**

**SELECT** @last\_contract **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Applicants **VALUES(NULL,**'Андрій'**,**'Хоменко'**,**'Дмитрович'**,**'2015-07-15 15:40:00'**,**'D:/abit.jpg'**,**@last\_applicant\_location**,**'+380965061435'**,**

'1997-10-22'**,**@last\_school**,**@last\_passport**,**@last\_school\_certification**,**'Петров П.П'**,**@last\_contract**,**5.0**,**@last\_zno\_certification**,**

**FALSE,**14561**,**1**);**

**SELECT** @last\_applicant **:=** LAST\_INSERT\_ID**();**

**INSERT** **INTO** Assertions **VALUES(NULL,**@last\_applicant**,**'2015-07-15 15:43:00'**,**@last\_faculty**,**@last\_specialty**,**4**,TRUE,TRUE);**